Speedup Your Analytics: Automatic Parameter Tuning for Databases and Big Data Systems

Jiaheng Lu,Yuxing Chen
University of Helsinki
first.last@helsinki.fi

Herodotos Herodotou
Cyprus Univ. of Technology
first.last@cut.ac.cy

Shivnath Babu
Duke University
shivnath@cs.duke.edu

ABSTRACT

Database and big data analytics systems such as Hadoop and Spark have a large number of configuration parameters that control memory distribution, I/O optimization, parallelism, and compression. Improper parameter settings can cause significant performance degradation and stability issues. However, regular users and even expert administrators struggle to understand and tune them to achieve good performance. In this tutorial, we review existing approaches on automatic parameter tuning for databases, Hadoop, and Spark, which we classify into six categories: rule-based, cost modeling, simulation-based, experiment-driven, machine learning, and adaptive tuning. We describe the foundations of different automatic parameter tuning algorithms and present pros and cons of each approach. We also highlight real-world applications and systems, and identify research challenges for handling cloud services, resource heterogeneity, and real-time analytics.

PVLDB Reference Format:
Jiaheng Lu, Yuxing Chen, Herodotos Herodotou, and Shivnath Babu. Speedup Your Analytics: Automatic Parameter Tuning for Databases and Big Data Systems. *PVLDB*, 12(12): 1970-1973, 2019.
DOI: https://doi.org/10.14778/3352063.3352112

1. MOTIVATION

The continuous growth of the World Wide Web, Internet of Things (IoT), E-commerce, and other applications are generating massive amounts of ever-increasing raw data every day. Data analytics platforms, including parallel and distributed database systems as well as large-scale data processing systems (e.g., *Hadoop MapReduce* and *Spark*), have emerged to assist with the Big Data challenge, i.e., to efficiently collect, process, and analyze massive volumes of heterogeneous data. Achieving good and robust system performance at such scale is the foundation to successfully performing timely and cost-effective analytics. However, system performance is directly linked to a vast array of configuration parameters, which control various aspects of system execution, ranging from low-level memory settings and thread counts to higher-level decisions like scheduling and resource management. Improper settings of configuration parameters are shown to have detrimental effects on the overall system performance and stability [9, 13].

The use of automated configuration parameter tuning techniques is a promising, yet challenging, approach to optimizing system performance. The major challenges include three aspects as follows: (i) **Large and complex parameter space:** Database systems often have hundreds of tuning knobs [24], while Hadoop and Spark have around 200 configurable parameters each [15]. To make matters worse, some parameters might affect the performance of different queries/jobs in different ways, while certain groups of parameters may have dependent effects (i.e., a good setting for one parameter may vary based on the setting of another parameter) [9, 13]. (ii) **System scale and complexity:** As data analytics platforms have grown in scale and complexity, system administrators may need to configure and tune hundreds to thousands of nodes, some provisioned with different CPU, storage, memory, and network technologies. In addition, executing MapReduce or Spark workloads with iterative stages and tasks in parallel or serial makes it challenging to observe and model workload performance [10]. (iii) **Lack of input data statistics:** Tuning database parameters for accelerating search queries require previous statistics or informational logs, which may not be available, especially for ad-hoc queries [9]. As for MapReduce and Spark applications, data statistics are rarely available since data often resides in semi- or un-structured files and is opaque until it is accessed [12].

There has been a significant amount of research works addressing this problem by providing self-configuring features in database systems (e.g. [7, 9, 29]), Hadoop MapReduce (e.g. [13, 15]), and Spark (e.g. [10, 25]). This tutorial will perform a comprehensive study of existing parameter tuning approaches, which tackle various challenges towards high resource utilization, fast response time, and cost-effectiveness. Due to the various challenges and scenarios addressed, different strategies and approaches are proposed accordingly. We classify these approaches into six main categories: rule-based, cost modeling, simulation-based, experiment-driven, machine learning, and adaptive tuning. These approaches will be analyzed in depth and compared within the context of database systems, Hadoop MapReduce, and Spark. A summary of the strengths and weaknesses of each approach is provided in Table 1.

Proceedings of the VLDB Endowment, Vol. 12, No. 12
ISSN 2150-8097.
DOI: https://doi.org/10.14778/3352063.3352112

Table 1: Strengths and weaknesses of the various approaches for automatic parameter tuning

Approach	Strengths	Weaknesses
Rule-based	• Do not require extra specialized software • Some parameters may be easy to adjust	• Time-consuming and labor-intensive process • Requires in-depth knowledge of system internals • Higher risk of performance degradation
Cost Modeling	• Very efficient for predicting performance • Good accuracy in many (basic) scenarios	• Hard to capture complexity of system internals & pluggable components (e.g., schedulers) • Models often based on simplified assumptions • Not effective on heterogeneous clusters
Simulation-based	• High accuracy in simulating dynamic system behaviors • Efficient for predicting fine-grained performance	• Hard to comprehensively simulate complex internal dynamics • Unable to capture dynamic cluster utilization • Not very efficient for finding optimal settings
Experiment-driven	• Find good settings based on real system test runs • Work across different system versions and hardware	• Very time consuming as they require multiple actual runs • Not cost effective for ad-hoc queries/applications
Machine Learning	• Ability to capture complex system dynamics • Independence from system internals and hardware • Learning based on real observations of system performance	• Require large training sets, which are expensive to collect • Training from history logs leads to data under-fitting • Typically low accuracy for unseen queries/applications • Hard to choose the proper model
Adaptive	• Find good settings based on real test runs on real systems • Able to adjust to dynamic runtime status • Work well for ad-hoc queries/applications	• Only apply to long-running queries/applications • Inappropriate configuration can cause issues (e.g., stragglers) • Neglect efficient resource utilization in the whole system

In this tutorial, we will first provide an overview and motivating examples of parameter tuning on the database and big data systems. Next, we will introduce the six categories to classify the existing tuning approaches, present the essential characteristics of each category, and discuss their respective strengths and weaknesses. Finally, we will highlight real world applications and systems for automatic parameter tuning, and identify research challenges to handle cloud services, resource heterogeneity, and real-time analytics.

To the best of our knowledge, this is the first tutorial to discuss the state-of-the-art research works and industrial trends in the context of parameter tuning. We have identified few tutorials (e.g., [5,6]) on automatic database tuning, which are mainly from VLDB and ICDE. However, they mainly focus on optimizing query execution plans or other higher level aspects such as index and materialized view creation on databases. This tutorial, on the other hand, focuses on the state-of-the-art works on parameter tuning, which optimizes the performance of the entire system as a whole. In addition, this tutorial covers the automatic tuning for popular big data analytic platforms, namely Hadoop and Spark. The slides of this tutorial are available online [16].

2. COVERED TOPICS

2.1 Background and Classification

The performance benefits of tuning are well-known in the industry, sometimes measured in orders of magnitude of improvement [24], while bad configurations (or misconfiguration) can lead to significantly degraded performance [27]. A significant amount of research has been performed over the last decade for automating parameter tuning in database and large-scale data processing systems. We classify these approaches into six categories:

1. **Rule-based** approaches assist users with tuning system parameters based on the experience of human experts, online tutorials, or tuning instructions. They usually require no models and are suitable for quickly bootstrapping the system.
2. **Cost modeling** approaches build efficient performance prediction models by using statistical cost functions via a deep understanding of system internals. No or few experimental logs are required to establish the model.
3. **Simulation-based** approaches build performance prediction models based on modular or complete system simulation, enabling users to simulate an execution under different parameter settings or cluster resources.
4. **Experiment-driven** approaches execute an application, i.e., an experiment, repeatedly with different parameter settings, guided by a search algorithm and the feedback provided by actual runs.
5. **Machine learning** approaches establish performance prediction models by employing machine learning methods. They typically consider the complex system as a whole and assume no knowledge of system internals.
6. **Adaptive** approaches tune configuration parameters adaptively while an application is running, meaning that they can adjust the parameter settings as the environment changes. They enable tuning of ad-hoc applications.

These approaches will be analyzed in depth and compared during the tutorial within the context of database systems, Hadoop MapReduce, and Spark.

2.2 Parameter Tuning on Database Systems

Several configuration parameters (e.g., buffer cache size, deadlock timeout) can significantly affect the performance of a DBMS. Several past approaches have addressed the general issue of parameter tuning, each trying to resolve one or more of the following specific problems: (i) avoiding error-prone configuration settings [23, 26]; (ii) ranking parameters based on their impact on system performance [11, 29]; (iii) profiling queries to collect useful log information for later prediction and use [28]; (vi) predicting the database or workload performance under hypothetical resource or parameter changes [1]; and (v) recommending and tuning parameter values to achieve objective goals [4]. Table 2 compares selected parameter tuning approaches in terms of their methodology, supported parameters, and target problems.

2.3 Parameter Tuning on Hadoop MapReduce

An early comparative study between Hadoop MapReduce and two parallel database systems revealed that Hadoop was slower by a factor of 3.1 to 6.5 in executing a variety of data-intensive analytical workloads [18, 21]. Motivated by

Table 2: An overview comparison of selected parameter tuning approaches for a DBMS

Category	Approach	Methodology	Parameters	Target Problems
Rule-based	SPEX [27]	Constraint inference	Several parameters	Avoid error-prone configs
	Tianyin [26]	Configuration navigation	Several parameters	Ranking the effects of parameters
Cost Modeling	STMM [22]	Cost-benefit analysis	Memory parameters	Tuning, Recommendation
Simulation-based	Dushyanth [17]	Trace-based simulation	CPU, memory, I/O	Prediction
	ADDM [8]	DAG model & simulation	CPU, I/0, DB locks	Profiling, Tuning
Experiment-driven	SARD [7]	P&B statistical design	Several parameters	Ranking the effects of parameters
	Shivnath [3]	Adaptive sampling	Several parameters	Profiling, Tuning
	iTuned [9]	LHS & Guassian Process	Several parameters	Profiling, Tuning
Machine Learning	Rodd [19]	Neural Networks	Memory parameters	Tuning, Recommendation
	OtterTune [24]	Guassian Process	Several parameters	Tuning, Recommendation
Adaptive	COLT [20]	Cost Vs. Gain analysis	Few parameters	Profiling, Tuning

these results, two performance studies [2, 14] conducted in-depth analyses of Hadoop in order to determine the most important factors and configuration parameters that affect its performance. Both studies concluded that by carefully tuning these factors and parameters, the overall performance of Hadoop can be dramatically improved and be more comparable to that of parallel database systems. These results stimulated a plethora of work in automatically tuning configuration parameters, which control various aspects of MapReduce job behavior (e.g., task concurrency, memory allocation, I/O performance). Specifically, we have identified over 40 highly-cited approaches (e.g., [13, 15]) spanning our six categories and published within the last 10 years.

2.4 Parameter Tuning on Spark

Spark is now one of the most prevalent large-scale data processing platforms, aiming to speed up large-scale data analytics in a broad spectrum of applications, such as training machine learning model and processing streaming data. Tuning Spark system performance is essential since fast and efficient performance leads to time-saving and high cluster-resource utilization; thus, to cost-effectiveness [25]. Spark performance is controlled by over 200 parameters from which about 30 can have a significant impact on job performance. These parameters mainly affect some aspects of execution and allocation of computing resources, such as CPU, memory, and network. In this part of the tutorial, we will present an in-depth analysis of over 15 approaches published in the last 4 years (e.g., [10, 25]), which can be beneficial for not only researchers who are doing relevant research work but also engineers who tune the system in production.

2.5 Open Problems and Challenges

In the last part of the tutorial, we focus on open challenges that must be addressed to ensure the success of automatic parameter tuning, especially when taking into account the growth of scale and complexity of big data analytics systems. The main areas to be discussed involve: (1) *Heterogeneity*: Tuning over heterogeneous hardware and software. (2) *Cloud computing*: Decision making in resource provisioning and scheduling with multiple tenants. (3) *Real-time analytics*: New challenges arise in such settings due to different architecture and low-latency response requirements. In each of these areas, we briefly overview partial/preliminary solutions and discuss the various challenges involved. We expect that this part will spark in-depth and interesting discussions.

3. TUTORIAL ORGANIZATION

The tutorial is planned for 1.5 hours and will have the following structure:

Motivation (5'). We motivate the need for automatic parameter tuning with several applications/scenarios in the era of Big Data and cloud computing.

History and classification (10'). We introduce the history and classification of parameter tuning approaches.

Parameter tuning on Databases, Hadoop, and Spark (55'). We introduce key approaches to tune performance on database systems, Hadoop MapReduce, and Spark. We compare the solutions in various tuning categories.

Applications of automatic parameter tuning (10'). We discuss some real applications and systems for automatic tuning, such as Self-driving Oracle Database, Self-tuning DB2 and Unravel platform.

Open problem and challenges (10'). We conclude with a discussion of open problems and challenges for parameter tuning.

4. GOALS OF THE TUTORIAL

4.1 Learning Outcomes

The main learning outcomes of this tutorial are as follows: (1) Motivation, classification and historical evolution of automatic parameter tuning approaches. (2) An overview of tuning approaches used by the current database and big data platforms including rule-based, cost modeling, simulation-based, experiment-driven, machine learning, and adaptive approaches. (3) Comparison of features, strengths, and applications of tuning approaches. (4) A discussion of research challenges and open problems of parameter tuning.

4.2 Intended Audience

This tutorial is intended for a wide scope of audience ranging from academic researchers to industrial data scientists that want to understand the impact of parameters on performance in big data analytics systems. Also, this tutorial can help not only motivated researchers and developers to select new topics and contribute their expertise on automatic parameter tuning, but also new developers and students to quickly build a comprehensive overview and grasp the latest trends and state-of-the-art techniques in this field.

Basic knowledge in parameter configuration in databases or big data systems is sufficient to follow the tutorial. Some background in cloud/cluster resource scheduling, performance

tuning, and basic machine learning techniques would be useful but not necessary.

5. SHORT BIBLIOGRAPHIES

Jiaheng Lu is an Associate Professor at the University of Helsinki, Finland. His main research interests lie in the big data management and database systems, and specifically in the challenge of efficient data processing from real-life, massive data repositories and the Web. He has written four books on Hadoop and NoSQL databases, and more than 70 journal and conference papers published in SIGMOD, VLDB, TODS, TKDE, etc.

Yuxing Chen is a doctoral student at the University of Helsinki. His research topics are parameter tuning on big data systems and multi-model query optimization.

Herodotos Herodotou is an Assistant Professor at the Cyprus University of Technology. His research interests are in large-scale data processing systems, database systems, and cloud computing. In particular, his work focuses on automated performance tuning of both centralized and distributed data-intensive computing systems. His Ph.D. dissertation work on the Starfish platform received the ACM SIGMOD Jim Gray Doctoral Dissertation Award Honorable Mention as well as the Outstanding Ph.D. Dissertation Award in Computer Science at Duke.

Shivnath Babu is the CTO at Unravel Data Systems and an Adjunct Professor at Duke University. His research focuses on ease-of-use and manageability of data-intensive systems, automated problem diagnosis, and cluster sizing for applications running on cloud platforms. Shivnath cofounded Unravel to solve the application management challenges that companies face when they adopt systems like Hadoop and Spark. Unravel originated from the Starfish platform built at Duke, which has been downloaded by over 100 companies. Shivnath has won a US National Science Foundation CAREER Award, three IBM Faculty Awards, and an HP Labs Innovation Research Award.

6. REFERENCES

[1] D. V. Aken, A. Pavlo, G. J. Gordon, and B. Zhang. Automatic Database Management System Tuning through Large-scale Machine Learning. In *SIGMOD*, pages 1009–1024. ACM, 2017.

[2] S. Babu. Towards Automatic Optimization of MapReduce Programs. In *SoCC*, pages 137–142. ACM, 2010.

[3] S. Babu, N. Borisov, S. Duan, H. Herodotou, and V. Thummala. Automated Experiment-Driven Management of (Database) Systems. In *HotOS*. USENIX Association, 2009.

[4] L. Cai, Y. Qi, W. Wei, J. Wu, and J. Li. mrMoulder: A Recommendation-based Adaptive Parameter Tuning Approach for Big Data Processing Platform. *Future Generation Computer Systems*, 93:570–582, 2019.

[5] S. Chaudhuri, B. Dageville, and G. M. Lohman. Self-Managing Technology in Database Management Systems. In *VLDB*, page 1243, 2004.

[6] S. Chaudhuri and G. Weikum. Foundations of Automated Database Tuning. In *ICDE*, page 104. IEEE, 2006.

[7] B. K. Debnath, D. J. Lilja, and M. F. Mokbel. SARD: A Statistical Approach for Ranking Database Tuning Parameters. In *ICDE Workshop*, pages 11–18. IEEE, 2008.

[8] K. Dias, M. Ramacher, U. Shaft, V. Venkataramani, and G. Wood. Automatic Performance Diagnosis and Tuning in Oracle. In *CIDR*, pages 84–94, 2005.

[9] S. Duan, V. Thummala, and S. Babu. Tuning Database Configuration Parameters with iTuned. *PVLDB*, 2(1):1246–1257, 2009.

[10] A. Gounaris, G. Kougka, R. Tous, C. T. Montes, and J. Torres. Dynamic Configuration of Partitioning in Spark Applications. *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, 28(7):1891–1904, 2017.

[11] Á. B. Hernández, M. S. Pérez, S. Gupta, and V. Muntés-Mulero. Using Machine Learning to Optimize Parallelism in Big Data Applications. *Future Generation Computer Systems*, 86:1076–1092, 2018.

[12] H. Herodotou and S. Babu. Profiling, What-if Analysis, and Cost-based Optimization of MapReduce Programs. *PVLDB*, 4(11):1111–1122, 2011.

[13] H. Herodotou, H. Lim, G. Luo, N. Borisov, L. Dong, F. B. Cetin, and S. Babu. Starfish: A Self-tuning System for Big Data Analytics. In *CIDR*, volume 11, pages 261–272, 2011.

[14] D. Jiang, B. C. Ooi, L. Shi, and S. Wu. The Performance of MapReduce: An In-depth Study. *PVLDB*, 3(1-2):472–483, 2010.

[15] S. Kadirvel and J. A. Fortes. Grey-box Approach for Performance Prediction in Map-Reduce based Platforms. In *ICCCN*, pages 1–9. IEEE, 2012.

[16] J. Lu, Y. Chen, H. Herodotou, and S. Babu. Presentation slides. `https://www.helsinki.fi/en/researchgroups/unified-database-management-systems-udbms/tutorial/vldb-2019-tutorial`, 2019.

[17] D. Narayanan, E. Thereska, and A. Ailamaki. Continuous Resource Monitoring for Self-predicting DBMS. In *MASCOTS*, pages 239–248. IEEE, 2005.

[18] A. Pavlo, E. Paulson, A. Rasin, et al. A Comparison of Approaches to Large-Scale Data Analysis. In *SIGMOD*, pages 165–178. ACM, 2009.

[19] S. F. Rodd and U. P. Kulkarni. Adaptive Tuning Algorithm for Performance Tuning of Database Management System. *IJCSIS*, 8(1):121–124, 2010.

[20] K. Schnaitter, S. Abiteboul, T. Milo, and N. Polyzotis. COLT: Continuous On-line Tuning. In *SIGMOD*, pages 793–795. ACM, 2006.

[21] J. Shi, J. Zou, J. Lu, Z. Cao, S. Li, and C. Wang. MRTuner: A Toolkit to Enable Holistic Optimization for MapReduce Jobs. *PVLDB*, 7(13):1319–1330, 2014.

[22] A. J. Storm, C. Garcia-Arellano, S. S. Lightstone, Y. Diao, and M. Surendra. Adaptive Self-tuning Memory in DB2. In *VLDB*, pages 1081–1092, 2006.

[23] Z. Tan and S. Babu. Tempo: Robust and Self-Tuning Resource Management in Multi-tenant Parallel Databases. *PVLDB*, 9(10):720–731, 2016.

[24] D. Van Aken, A. Pavlo, G. J. Gordon, and B. Zhang. Automatic Database Management System Tuning through Large-scale Machine Learning. In *SIGMOD*, pages 1009–1024. ACM, 2017.

[25] S. Venkataraman, Z. Yang, M. J. Franklin, et al. Ernest: Efficient Performance Prediction for Large-Scale Advanced Analytics. In *NSDI*, pages 363–378. USENIX, 2016.

[26] T. Xu, L. Jin, X. Fan, Y. Zhou, S. Pasupathy, and R. Talwadker. Hey, You Have Given me too Many Knobs!: Understanding and Dealing with Over-designed Configuration in System Software. In *ESEC*, pages 307–319. ACM, 2015.

[27] T. Xu, J. Zhang, P. Huang, J. Zheng, et al. Do Not Blame Users for Misconfigurations. In *SOSP*, pages 244–259. ACM, 2013.

[28] N. Zacheilas, S. Maroulis, T. Priovolos, V. Kalogeraki, and D. Gunopulos. Dione: A Framework for Automatic Profiling and Tuning Big Data Applications. In *ICDE*, pages 1637–1640. IEEE, 2018.

[29] B. Zhang, D. Van Aken, J. Wang, T. Dai, S. Jiang, et al. A Demonstration of the OtterTune Automatic Database Management System Tuning Service. *PVLDB*, 11(12):1910–1913, 2018.